연속미디어를 위한 스케쥴링 알고리즘

유명련[†] · 안병철[††]

요 약

연속적인 미디어 특성을 가진 데이터들은 시간적인 제약조건을 가진다. 일반적인 실시간 스케쥴링 알고리즘은 연속적인 미디어 특성을 고려하지 아니하므로 멀티미디어 스케쥴링에 적절하지 않다.

스트라이드 스케쥴러를 기본으로 하여 설계된 비율조정 비례지분 스케쥴러는 연속 미디어의 시간 제약적인 특성을 고려한 스케쥴링 알고리즘이다. 일반적인 태스크를 위해 설계된 스트라이드 스케쥴러는 자원 할당에 있어서 공정성과 예측가능성을 보장한다. 비율조정 비례지분 스케쥴러에서 연속 미디어의 특성을 고려하기 위해 도입한 비율조정기는 태스크가 지분보다 더 많은 자원을 할당 받지 않도록 조정을 해 준다. 그러나 엄격한 비율조정기로 인해 비율조정 비례지분 스케쥴러는 자원 할당의 공정성을 보장하지 못하다. 본 논문에서 제시하는 수정된 비례지분 스케쥴러는 연속성, 시간 제약성과 같은 연속 미디어의 특성을 고려한다.

본 논문에서 제시하는 스케쥴링 알고리즘은 자원 할당의 공정성을 유지하여 과부하시에 성능이 점진적으로 저하됨을 보였고, 비율조정 비례지분 스케쥴러보다 문맥교환에 있어 좀 더 나은 성능을 보였다.

A Scheduling Algorithm for Continuous Media

Myung-Ryun Yoo[†] and Byoung-Chul Ahn[††]

ABSTRACT

Since continuous media such as video and audio data are displayed within a certain time constraint, their computation and manipulation should be handled under limited condition. Traditional real-time scheduling algorithms could not be directly applicable, because they are not suitable for multimedia scheduling applications which support many clients at the same time.

Rate Regulating Proportional Share Scheduling Algorithm based on the stride scheduler is a scheduling algorithm considered the time constraint of the continuous media. The stride schedulers, which are designed to general tasks, guarantee the fairness of resource allocation and predictability. The key concept of RRPSSA is a rate regulator which prevents tasks from receiving more resource than its share in a given period. But this algorithm loses fairness which is a strong point of the stride schedulers, and does not show graceful degradation of performance under overloaded situation.

This paper proposes a new modified algorithm, namely Modified Proportional Share Scheduling Algorithm considering the characteristics of multimedia data such as its continuity and time dependency. Proposed scheduling algorithm shows graceful degradation of performance in overloaded situation and it reduces the scheduling violations up to 70% by maintaining the fair resource allocation. The number of context switching is 8% less than RRPSSA and the overall performance is increased.

1. 서 론

멀티미디어 시스템을 구성하는 데이터들 중 비디오와 오디오는 연속미디어(continuous media)로 분류된다. 연속미디어는 주어진 시간안에 처리가 완료되어 재생되어야 하는 특성을 가지므로 텍스트와 같은 이산미디어보다 제한된 처리조건을 가지게 된다. 전통적인 운영체제의 스케쥴러(scheduler)는 이러한 시간제약조건을 고려하여 설계되지 않았으므로 연속미디어를 지원하기에는 적합하지 않다[1,2].

† 정회원, 안동정보대학 인터넷 정보과 조교수
†† 영남대학교 공과대학 컴퓨터공학과 부교수

연속미디어를 효과적으로 다루기 위해 주기가 짧은 태스크(task)에게 주기가 긴 태스크보다 높은 우선순위를 부여해 주는 RM(Rate Monotonic)과 같은 정적인 스케쥴링 알고리즘과 마감시간에 가장 임박한 태스크에게 가장 높은 우선순위를 부여해 주는 EDF(Earliest Deadline First)와 같은 동적인 스케쥴링 알고리즘이 꾸준히 개선되어 왔다. RM이나 EDF 그리고 수정된 알고리즘들은 경성 실시간 시스템을 기본으로 하여 개발된 알고리즘들이다. 이러한 알고리즘들의 목표는 모든 태스크의 마감시간을 지키는 것과 항상 스케쥴링 가능하도록 하기 위해 입장제어(admission control)를 둔다는 것이다[3,4].

경성 실시간 시스템에서의 스케쥴링 알고리즘은 과부하상태에서 시스템이 예측 불가능하게 동작하는 것을 방지하기 위해 엄격한 입장제어를 사용하고 있으므로 자원의 낮은 활용도의 원인이 될 수 있다. 멀티미디어 시스템에서는 연속미디어의 마감시간을 다소 어기는 것이 치명적인 결과를 초래하지는 않으며, 각 미디어의 서비스 품질(Quality of Service)을 만족시킬 수 있는 범위 내에서 약간의 마감시간을 어기는 것은 허용되어질 수 있다. 그러므로, 멀티미디어 시스템에서의 연속미디어는 QoS(Quality of Service)를 이용하여 경성 실시간 시스템에서의 태스크보다 좀 더 유연하게 스케쥴링 될 수 있다[5].

연속미디어를 위한 또 다른 스케쥴링 방법으로 비율조정 비례지분 스케쥴링 알고리즘(Rate Regulating Proportional Share Scheduling Algorithm)이 있다. RRPSSA[5]는 비례지분(proportional share) 방법에 기초를 둔 스트라이드(stride)알고리즘을 수정한 것이다. 스트라이드 알고리즘은 일반적인 태스크를 위해 설계되었으며, 자원 할당에 있어서 공정성과 시스템의 예측가능성을 유지한다[6,7]. RRPSSA는 연속적인 미디어를 스케쥴링하기 위해 스트라이드 알고리즘을 변형하여 연속미디어의 시간적인 특성, 즉, 태스크의 주기와 각 주기안에서 요구되어지는 실행시간을 반영하였다. 비율조정기는 RRPSSA의 가장 중요한 개념이라고 할 수 있다. 비율조정기는 태스크가 주어진 주기내에서 자원을 자신의 지분보다 더 많이 할당 받지 못하도록 한다[5]. 그러나 이 알고리즘은 엄격한 비율조정기로 인하여 스트라이드 알고리즘의 장점인 자원할당에서의 공정성을 유지하지 못하고, 과부하 상태에서 성능을 예측 할 수 없다.

이 논문에서는 스트라이드 알고리즘에 연속미디어의 시간적인 특성을 반영한 수정된 비례지분 스케쥴링 알고리즘(Modified Proportional Share Scheduling Algorithm)을 제시한다. MPSSA는 자원할당에 있어서 공정성을 유지하고 과부상태에서도 성능이 점진적으로 저하되며 RRPSSA보다 문맥교환의 횟수가 적다.

이 논문의 2장에서는 RRPSSA의 특징 및 장단점을 설명하고 개선점을 논한다. 3장에서는 MPSSA를 소개한다. 4장에서는 제안한 방법에 대한 성능을 측정하기 위해 수행한 모의실험 결과를 보이고, 5장에서 결론을 맺는다.

2. RRPSSA 스케쥴링

2.1 RRPSSA 스케쥴링 방법

스트라이드 스케쥴러를 기본으로 하는 RRPSSA는 연속미디어를 스케쥴링하기 위해 설계된 알고리즘이다. 연속미디어의 시간적인 특성을 반영하기 위해 한 쌍의 파라메타 (P, C)를 사용한다. 파라메타 P는 태스크의 주기를 나타내고, 파라메타 C는 태스크의 각 주기안에서 요구되어지는 계산시간을 의미한다.

알고리즘 진행과정에서 세 개의 파라메타 티켓(ticket), 스트라이드(stride), 패스(pass)가 사용된다. 티켓은 태스크의 주기에 대한 계산시간을 값으로 가지며 자원에 대한 권리를 나타낸다. 스트라이드는 티켓 값의 역수이고, 스케쥴러로부터 선택받는 간격을 나타낸다. 패스는 초기에 스트라이드와 같은 값을 가지고, 스케쥴러의 다음 선택을 위한 가상 시간 인덱스(virtual time index)를 나타낸다. 스케쥴러는 패스값이 가장 작은 태스크를 선택하고, 선택된 태스크의 패스는 스트라이드 값만큼 증가하게 된다[6,7].

RRPSSA의 가장 중요한 개념인 비율 조정기는 태스크가 한 주기안에서 자원을 자신의 지분보다 더 많이 할당 받지 못하도록 조정한다. 비율조정기는 다음 식(1)과 같이 나타낼 수 있다.

$$\sum_{i=start}^{current} \frac{Alloc_k(i)}{\Delta T} \le \frac{C_k}{P_k} \qquad (1)$$

식 (1)에서 사용한 기호는 아래와 같다.

$\Sigma Alloc_k(i)$: 시작부터 현재까지 태스크가 할당된

전체 수

T	: 시작부터 현재까지의 시간
C_k	: 태스크 k ($task_k$)의 계산 시간
P_k	: 태스크 k ($task_k$)의 주기

RRPSSA는 연속미디어를 스케줄링하기 위해 설계된 알고리즘으로 연속미디어의 시간적인 특성을 고려했으며, 저부하 상태에서 자원 할당의 공정성을 유지한다.

2.2 RRPSSA의 개선점

RRPSSA가 2.1절에서 언급한 장점을 가지고 있음에도 불구하고 연속미디어를 지원하기에는 다음과 같은 문제점이 있다.

첫째, 이 알고리즘은 과부하상태에서 성능이 점진적으로 저하되지 않는다. 스케줄러로부터 서비스 받지 못한 시간의 횟수면에 있어서는 공정성을 유지하는 것처럼 보이지만, 서비스 받아야 할 시간에 대한 서비스 받지 못한 시간의 비율면에 있어서는 공정성을 유지하지 못한다.

둘째, 이 알고리즘은 엄격한 비율조정기로 인하여 문맥교환에 따르는 오버헤드가 너무 크다. 문맥교환은 기계에 의존적인 요소이고, 전체 계산 시간에 비해 상대적으로 매우 작은 시간이므로 스케줄러 설계시 거의 무시되고 있다. 그러나 스케줄링 대상이 되는 태스크의 수가 많을 때 과도한 문맥교환은 쓰레싱(trashing)을 유발하고, 시스템의 성능을 저하시키게 된다.

위에서 언급한 두 가지의 단점을 보완하고 연속미디어를 스케줄링하기 위해 스케줄러 설계시 다음과 같은 사항들이 고려되어야 한다.

1) 과부하 상태에서도 성능이 점진적으로 저하되어야 한다.
2) 문맥교환 횟수를 줄인다.
3) 자원 활용율의 저하를 야기시키는 입장 제어를 사용하지 않는다.

3. MPSSA 스케줄링

3.1 자원 할당

MPSSA에서의 티켓은 할당받아야 할 자원에 대한 권리를 나타내고 식(2)로 표시된다. 식 (2)는 저부하 상황과 과부하 상황 모두를 고려하여 할당 받아야 할 자원의 비율을 나타낸다.

$$Ticket_k = \begin{cases} \dfrac{\frac{C_k}{P_k}}{\sum_{i=1}^{n}\frac{C_i}{P_i}} & if \sum_{i=1}^{n}\frac{C_i}{P_i} > 1 \\ \dfrac{C_k}{P_k} & if \sum_{i=1}^{n}\frac{C_i}{P_i} \le 1 \end{cases} \tag{2}$$

수식 (2)의 기호는 다음과 같다.

k	: 태스크 k ($task_k$)
C_k	: 태스크 k ($task_k$)의 계산 시간
P_k	: 태스크 k ($task_k$)의 주기

패스는 스케줄러의 다음 선택을 위한 가상 시간 인덱스를 나타낸다. 초기값으로 0을 가지며, 매 시간마다 다음 식(3)에 의해 변경된다.

$$Pass_k(t) = Ticket_k - \frac{\sum_{i=start}^{current} Alloc_k(i)}{\Delta T} \tag{3}$$

식 (3)의 부호는 다음과 같다.

t	: 현재 시간
$\Sigma Alloc_k(i)$	: 시작부터 현재까지 태스크가 할당된 전체 수
ΔT	: 시작부터 현재까지의 시간

3.2 태스크 선택

전 단계에서 선택된 태스크는 스케줄러에 대한 우선권을 가지고, 비율 조정기 식을 만족할 경우 스케줄러에 의해 다시 선택된다. 비율 조정기가 만족되지 않을 경우 패스 값이 가장 큰 태스크가 선택된다. 비율 조정기는 식 (4)와 같다.

$$Pass_k(t) > 0 \tag{4}$$

스케줄러에 대한 우선권 정책을 사용하여 문맥교환의 횟수를 감소시킬수 있다.

3.3 MPSSA의 흐름도

제안된 알고리즘의 흐름도는 다음 그림 1과 같다. 그림 1에서 태스크의 티켓과 패스는 태스크의 주

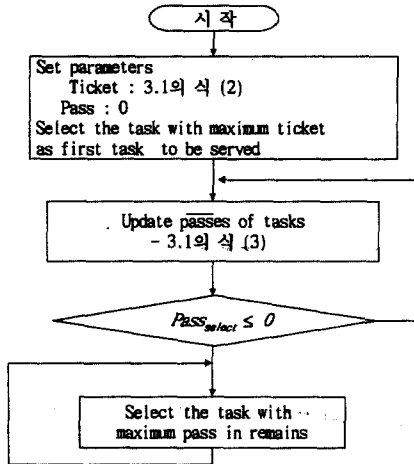



그림 1. MPSSA의 흐름도

기와 계산시간으로부터 결정된다. MPSSA의 스케줄러는 태스크의 티켓과 패스값을 비교하여 태스크를 선택한다. 스케줄링 되는 동안 3.2절에서 설명했듯이 문맥교환의 횟수를 감소시킬 수 있고, 식 (2)에서 보여지는 것처럼 과부하 상태에서도 자원 할당의 비율을 조정하여 성능이 점진적으로 저하된다.

(C, P)로 표시되는 세 가지의 태스크를 가지고 스케줄링 하는 예를 표 1에 설명 하였다. 태스크들은 각각 (1,5), (2,4), (2,6)으로 표시된다. 태스크들은 MPSSA에 의해 스케줄되고 스케줄된 결과를 다음 표1에 보였다. 표 1에서 태스크 A, B, C 는 각각의 패스값에 의해 선택되고, 스케줄러는 자원을 할당하는데에 있어 공정성을 유지하였다.

표 1. MPSSA 스케줄링의 예

티켓, A : B : C = 0.19 : 0.48 : 0.32				
시간	패		스	선택된 태스크
1	0	0	0	B
2	0.19	-0.52	0.32	C
3	0.19	-0.02	-0.18	A
4	-0.14	0.15	-0.01	B
5	-0.06	-0.02	0.07	C
6	-0.01	0.08	-0.08	B
7	0.03	-0.02	-0.01	A
8	-0.09	0.06	0.04	B
9	-0.06	-0.02	0.07	C
10	-0.03	0.04	-0.01	B
11	-0.01	-0.02	0.02	C
…	…	…	…	…

4. 모의 실험

MPSSA 알고리즘을 평가하기 위해 모의 실험을 하였다. 실험은 저부하 상태와 과부하 상태에서 이루어졌고 과부하 상태에서는 성능 저하 형태와 문맥교환의 수로 성능을 평가하였다

4.1 저부하 상태

저부하 상태에서는 문맥교환의 횟수면에서 MPSSA와 RRPSSA를 비교한다. 표 2는 저부하 상태에서 실험대상이 된 태스크들의 주기와 계산시간을 보여준다. 표 3는 실험 결과로 나타난 문맥교환의 횟수를 나타낸다.

표 2. 저부하 상태에서 스케줄링 대상이 된 태스크들

태스크	계산시간	주기
A	5	20
B	10	50
C	20	60

표 3에서 MPSSA 알고리즘을 사용했을 경우 RRPSSA보다 문맥교환의 횟수가 10%정도 감소함을 알 수 있다.

표 3. 1000퀀텀 시간에서의 문맥교환의 횟수

스케줄러	문맥교환 수
RRPSSA	604
MPSSA	545

4.2 과부하 상태

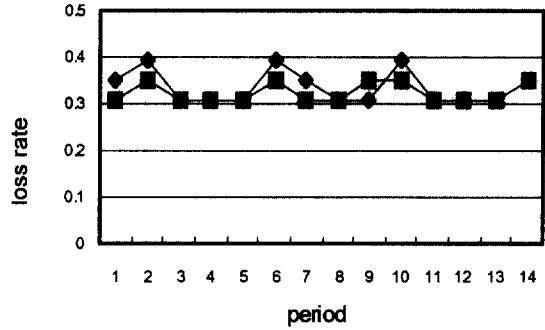
과부하상태에서 역시 MPSSA와 RRPSSA 를 비교한다. 문맥교환의 횟수와 성능저하의 형태면에서 비교를 하였다.

표 4는 스케줄링 대상이 된 태스크들의 주기와 계산시간을 나타낸다. 표 5에서는 RRPSSA와 MPSSA의 문맥교환 횟수를 비교하였다. 또한 그림 2에서는 성능저하의 형태를 나타내었다. 그림 2는 태스크 A,B,C,D의 손실 비율(loss rate)을 각각 도시하였다. 손실 비율은 태스크의 한 주기의 계산시간 동안 스케줄러로부터 서비스 받지 못한 시간의 비율이다. 즉, 손실 비율의 범위가 작다는 것은 손실 비율의 분산이 작다는 것을 의미한다.

표 4. 과부하 상태에서 실험 대상이 되는 태스크

태스크	계산시간	주기
A	23	100
B	68	150
C	35	100
D	23	70

표 5. 1000퀀텀 시간동안의 문맥교환 횟수

스케쥴러	문맥교환 수
RRPSSA	998
MPSSA	919

(a) 태스크 A

(b) 태스크 B

(c) 태스크 C

(d) 태스크 D

그림 2. 태스크의 손실 비율

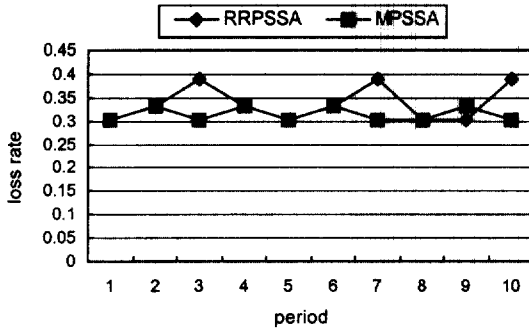
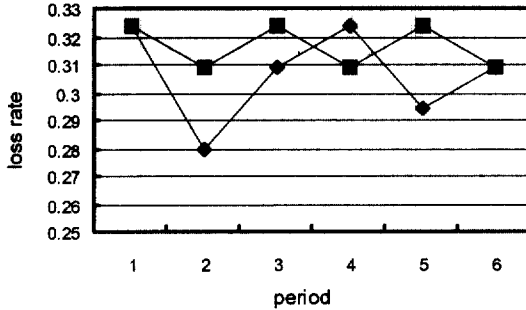
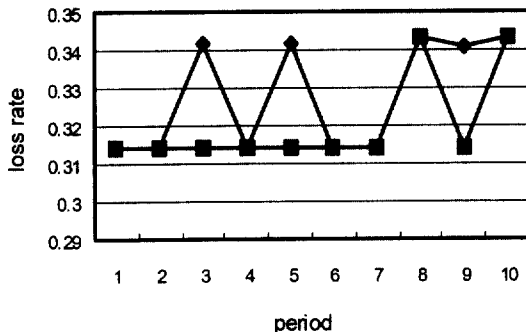



표 5에서 MPSSA의 문맥교환 횟수는 RRPSSA보다 8% 감소하였다. 그림 2에서 MPSSA에서의 손실 비율 범위가 RRPSSA에서 보다 작다. 표 6은 각 태스크의 손실 비율의 최대 값, 최소 값, 평균, 분산을 나타낸다. 표 6에서 MPSSA의 손실 비율의 분산이 RRPSSA보다 70%정도 작은 것을 알 수 있다. 이것은 MPSSA알고리즘을 사용했을 경우 RRPSSA를 사용했을 때 보다 과부하 상태에서 점진적으로 성능이 저하되고 있음을 증명한다.

표 6. 손실 비율 값

Loss Rate		A	B	C	D
Maximum	RRPSSA	0.391	0.324	0.343	0.391
	MPSSA	0.333	0.324	0.343	0.348
Minimum	RRPSSA	0.303	0.279	0.314	0.304
	MPSSA	0.303	0.309	0.314	0.304
Average	RRPSSA	0.338	0.306	0.328	0.332
	MPSSA	0.315	0.316	0.319	0.320
Variation	RRPSSA	0.00130	0.00025	0.00019	0.0012
	MPSSA	0.00021	0.00005	0.00013	0.0004

5. 결 론

이 논문은 연속 미디어를 위한 스케쥴링 알고리즘을 제시하였다. 모의실험 결과 다음과 같은 성능의 향상이 있었다.

1) MPSSA의 손실 비율이 RRPSSA보다 약 70% 정도 감소되었다. 이것은 과부하상태에서 성능이 훨씬 더 점진적으로 저하됨을 의미한다.

2) MPSSA의 문맥교환횟수가 RRPSSA보다 약

8%정도 줄었다. MPSSA에서는 불필요한 문맥교환에 따르는 오버헤드를 줄일 수 있다.

또한 MPSSA에서는 입장제어를 사용하지 않으므로 자원활용도를 높일 수가 있다.

본 논문에서 제시된 알고리즘을 실제 시스템에 구현하여 연속미디어를 이용한 성능을 평가하여 발전시킬 수 있다.

참 고 문 헌

[1] Ming-Syan Chen, Stream Conversion to Support Interactive Video Playout, *Proc. of the IEEE*, pp. 51, 1996

[2] Tsun-Ping J. Babak Hamidzadeh, Dynamic real time scheduling strategies for interative continuous media servers, *Miltimedia Systems 7*, pp. 91, 1999.

[3] L. Sha, R. Rajkumar, and S. S. Sathaye. rate monotonic scheduling theory, *Proc. of the IEEE*, pp.68, 1994

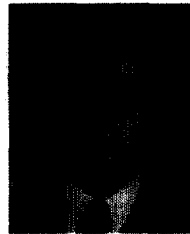
[4] C. M. Krishna, Kang G. Shin Real Time System, The *McGraw-Hill Companies, Inc.*

[5] Manhee Kim, Hyogun Lee, Joowon Lee, A Proportional Share Scheduler for Multimedia Applications, *Proc. of the IEEE*, pp. 4841997.

[6] Carl A. Waldspurger, William E. Weihl, Stride Scheduling: Deterministic Proportional Share Resource Management, *Technical Memorandum MIT/LCS/TM-528*, 1995

[7] Carl A. Waldspurger, Lottery and Stride Scheduling: Flexible Proportional Share Resource Management, *PhD thesis, MIT*, September, 1995

유 명 련

1994년 안동대학교 컴퓨터공학과 (공학사)
1996년 포항공과대학 정보통신대학원(공학석사)
1997년~현재 영남대학교 컴퓨터공학과 박사과정
1996년~현재 안동정보대학 인터넷 정보과 조교수
관심분야 : 운영체제, 멀티미디어

안 병 철

1976년 영남대학교 전자공학과 (공학사)
1986년 오레곤 주립대 전기 및 컴퓨터 공학과(공학석사)
1989년 오레곤 주립대 전기 및 컴퓨터 공학과(공학박사)
1978년~1984년 국방과학연구소 연구원
1989년~1992년 삼성전자 컴퓨터 부문 수석 연구원
1992년~현재 영남대학교 공과대학 컴퓨터공학과 부교수
관심분야 : 컴퓨터구조, 그래픽스, 멀티미디어 및 실시간 운영체제